Improving Robustness of Artificial Neural Networks Model Using Genetic Algorithm

Arshad Ahmad[1] **Chen Wah Sit[2]**

[1] *Laboratory of Process Control, Chemical Engineering Department, Faculty of Chemical and Natural Resources Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia. Tel: +60-7-5535610, Fax: +60-7-5581463, E-mail: arshad@fkkksa.utm.my*

2 Laboratory of Process Control, Chemical Engineering Department, Faculty of Chemical and Natural Resources Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia. Tel: +60-7-5535858, E-mail: wszeck@yahoo.com

Abstract

Artificial Neural Networks (ANN) has been widely accepted as process estimators due to its ability to capture complex relationships. However, experiences in implementing ANN estimators in research and industry have exposed some weaknesses that can be detrimental to the overall performance of plant operations. Among these, the issue of robustness is of particular importance. This paper proposes adaptation of network weights as means to improve robustness. Comparisons between GA approach and conventional backpropagation in adaptation of weights are made in on-line estimation and control of fatty acid composition in a distillation column. Significant improvements were obtained by the adaptive model especially model generalization perspective.

Keywords:

Artificial Neural Networks, Connection Weights, Genetic Algorithm, Inferential Estimation, Robustness

Introduction

Over the years, the application of Artificial Neural Networks (ANN) in the field of process industries is growing in acceptance. This is because ANN is capable of capturing process information in a black box manner. In fact, given sufficient input-output data, ANN is able to approximate continuous function to arbitrary accuracy. This has been proved in various fields such as pattern recognition, system identification, prediction, signal processing, fault detection and others [1].

In general, the development of a good ANN model depends on several factors [2]. The first factor is regarding the issue of process data. The quality of black box model is strongly influenced by the quality of data. Unless a good quality data is used in model fitting, the resulting model will not be adequate. The second factor is network architecture or model structure. Commonly, multilayer perceptron and its variances are widely used in process estimation. Different network architecture results in different estimation performance. The model size and complexity is another important factor. What is required is a parsimonious model. This is because a small network may not able to represent the real situation due to its limited capability, while a large network may overfit noise in the training data and fail to have good generalization ability. Finally, the quality of a process model is also strongly determined by network training. This stage is essentially an identification of model parameters that fits the given data.

This paper focuses on the last issue. The aim is to improve the estimation ability of ANN regardless of the network architecture. Until today, many researchers still prefer use gradient search method – Backpropagation (BP) in training ANN. Among all the backpropagation methods, Levenberg -Marquardt (LM) algorithms is the most widely used. The advantages of this gradient-based technique are its efficient implementations, good at fine-tuning and faster convergence compared to other methods. Unfortunately, BP is a local search method and when applied to complex nonlinear optimization problems, can sometimes result in inconsistent and unpredictable performances. Modifications proposed by various researchers provided only little improvement. This could be due to the fact that searching of optimal weights is strongly dependent on initial weights. If these initial weights are located near local minima, the algorithm would be trapped.

Several different attempts have been proposed by various researchers to alleviate this training problem. These include imposing constraints on the search space, restarting training at many random points, adjusting training parameter and restructuring the ANN architecture [3]. However, some approaches are problem-specific and not well accepted. So far, different researchers prefer different methodologies. One of the more promising directions in conjunction with improving ANN ability is by introducing adaptation of network training using Genetic Algorithms (GA).

Unlike BP, GA is a global search algorithm based on the principle "survival of fittest". GA simultaneously searches

ISBN:983-2643-15-5

for solutions in several regions, thus increasing the probability of global convergence. Besides, GA requires no knowledge or gradient information about the search space. Furthermore, since it is impossible to formulate an a priori exact model of the system, a more practical approach is offline set up a rough model, followed by online update of the model using GA. In this way, ANN-GA that results from the merging of GA and ANN will gain adaptability to dynamic environment. In other words, the ANN-GA model developed should be more robust to dynamic nonlinearity of the process involved.

In this work, an ANN model is used for inferential estimation of product composition in a distillation column. The aim is to address an issue in process industries, i.e., the lack of on-line measurement of product quality. The use of on-line measurements for product quality variables has been limited due to large measurement delay, the need for frequent maintenance as well as high capital and operating costs. In order to adapt to changing market conditions while maximizing profit, the needs for robust and accurate inferential estimators for controlling the product quality variable becomes paramount [4]. For this reason, this work introduces evolution of connection weights in ANN using GA as means of improving robustness of the resulting estimators.

Artificial Neural Networks

ANN is collections of mathematical models that emulate some of the observed properties of biological nervous systems. ANN mimics human learning and performs mapping from an input space to an output space. In general, ANN is made up of individual interconnected simple processing elements called neurons, arranged in a layered structure to form a network that allows parallel computation. Architecture of a general ANN is illustrated in Figure 1.

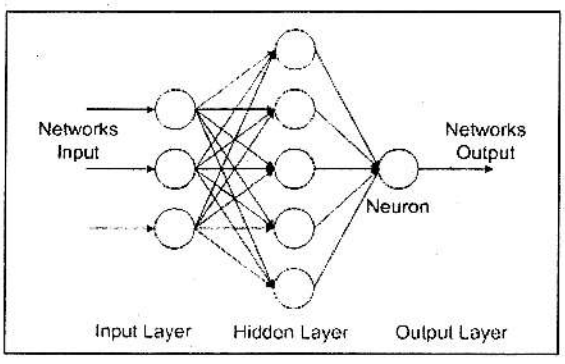


Figure 1 – Architecture of Artificial Neural Networks

An artificial neuron roughly replicates the operations and connectivity of their biological counterparts that are highly interconnected. An example of artificial neuron is illustrated in Figure 2. The neuron input, x_i, is multiplied by the corresponding weight factor, w_i, before being sent to the neuron. This is followed by performing summation of all input in the neuron body. An internal bias, b is also introduced to enhance performance of the network. The result is then passed through a nonlinear activation function or transfer function to obtain the output y:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{1}$$

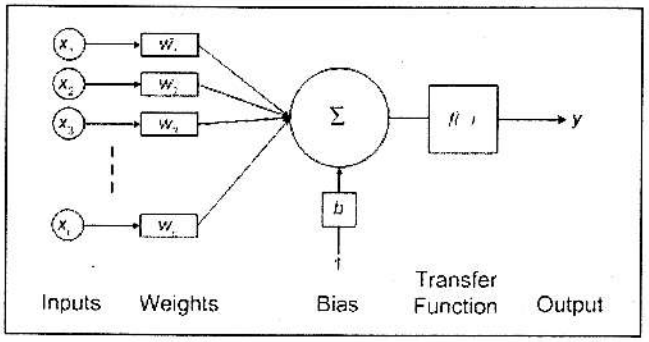


Figure 2 – Structure of Neuron Model

An important feature of this neuron model is a transfer function, $f(\ )$ which can perform nonlinear calculation. Various types of nonlinearity are possible such as log sigmoid, hyperbolic tangent sigmoid, hard limit, saturating linear and pure linear function.

ANN can perform a human-like reasoning, learns the attitude and stores the relationship of the processes on the basis of a representative data set that has already exists. Therefore, generally speaking, the ANN does not need much of a detailed description or formulation of the underlying process. Depending on the structure of the network, weights are adjusted in order to fit a series of inputs and output data. When the weight of a particular neuron is updated it is said that the neuron is learning or undergo training. Since the connecting weights are not related to some physical identities, the approach is considered as a black-box model. In summary, training of an ANN is an optimization problem where convergence to global minimum is desired.

In the recent years, ANNs have been extensively studied in academia as process models and controllers [5, 6, 7, 8]. The applications of ANN are increasing in process industry since it shown the effectiveness in process estimation. Although ANNs have been shown to be effective in solving difficult problems, specific guidance of design and implementation are still lacking [9]. Among the drawbacks of ANN is the use of gradient descent methods resulted in problems in the training of the network. This has motivated researchers to come out with idea of merging evolutionary algorithm (EA) into neural networks [10, 11].

Genetic Algorithms

GA belongs to a class of population based stochastic search algorithms known as Evolutionary Algorithms (others include Evolution Strategies, Evolutionary Programming and Genetic Programming). These algorithms model some natural phenomena of genetic inheritance and Darwinian strife for survival. In general, EAs are applicable to a wide

ISBN:983-2643-15-5

range of problem in learning and optimization. EAs are particularly useful for dealing with large complex problems that generate many local optima.

The evolution process implemented by GA can be interpreted as result of the interplay between the creation of new genetic information and its evaluation and selection. In life, an individual within a population is affected by other individuals as well as the surrounding environment. Normally, the better individuals have higher probability to live longer and to generate offspring. They in turn inherit the parental genetic information.

The evolution process involved in genetic algorithms are carried out by three main genetic operators: reproduction, crossover and mutation. The alterations of these genetic operators are based on probability defined earlier. The direct reproduction operator simply copies a member from the parental population to the next generation. The genetic operation of crossover takes two members of the population and combines their genetic information to create new offspring (children) as shown in Figure 3. The role of this operator is to allow the construction of new individuals from existing ones, thus enabling new regions of the solution space to be search.

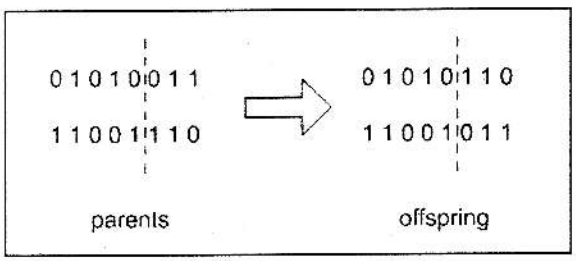


Figure 3 – A Typical Single Point Crossover for GA

The third operator is mutation. In this case, random alterations to the individual genetic coding are made to maintain population diversity, hence reducing the risk of premature convergence. This operation can be illustrated in Figure 4.

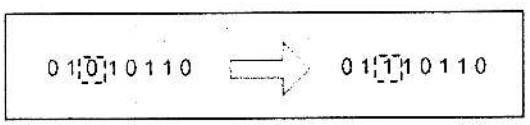


Figure 4 – A Typical Binary Mutation for GA

Although GA was originally developed for computer science research field, their applications have extended to process modelling and estimation of chemical processes. The application of GA in system identification can broadly be classified into two distinct areas: offline design and online optimization [12]. With the ability of global, parallel and stochastic search method, the idea of applying GA in ANN training seems to be advantageous.

ANN-GA model refer to a special class of ANN in which evolution is another fundamental form of adaptation in addition to learning. Both models are inspired by nature, but while ANN are concerned with learning of an individual (phenotypic learning); GA deal with learning of adaptation to a changing environment (genotypic learning) [13]. Intersection of both techniques is believed will improve the performance of ANN in process estimation.

The evolution steps of the ANN-GA are shown in Figure 5. During this evolution, the architecture of an ANN is redefined and fixed.

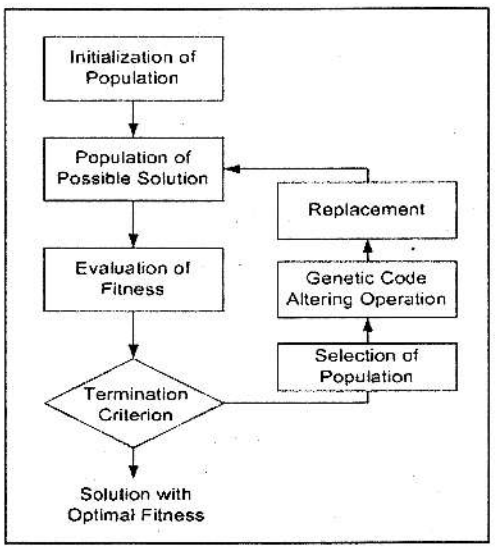


Figure 5– A General Genetic Algorithm Flowsheet

The first step in the evolution of ANN involves generation of initial population by decoding each individual in the current generation into connection weights. Each ANN with the decoded connection weights is trained by a predefined learning rule. The fitness of each individual is then computed. Here fitness represents the error due to the training process result and is represented as a positive real number, the higher the number, the better the individual.

The following step is the selection of fitter individuals in a population. This is aimed at producing fitter offspring. Numerous selection processes are available. There are three major selection schemes for GA: Roulette wheel selection, Rank-based selection and Tournament selection. Some individuals may be selected more than once for genetic operator alteration.

Selected populations shall then undergo genetic alteration to transfer their good genetic material to next generations. The replacement process comes immediately after the generation of new offsprings. The choice can be made either from the set of offspring only of from both sets of offspring and parents. In either case, the replacement procedure can be deterministic or stochastic. These evolutions are stopped according to generation counter or the number of function evaluations.

ISBN:983-2643-15-5

Process Description and Plant Simulation

In this work, an ANN model trained using LM (ANN-LM) and the one trained by GA (ANN-GA) are tested on Fatty Acid Fractionation Plant of Pan Century Oleochemicals Company (PCOC). The main function of the Fractionation Plant is to separate either palm kernel fatty acid (PKFA) or palm stearine hydrogenated fatty acid (PSHFA) into their fatty acid components noted as C14, C16, C18 and so on. The Fractionation Plant consists of the five systems connected in series. The separation of crude fatty acid into its components is achieved through the five column systems. These systems include:

- Precut Column System
- Lights Cut Column System
- Middle Cut Column System
- Still System
- Residue Still System

This research focuses only on the light cut column. The light cut column is mainly used to separate C-12 from the others. Similar to other distillation column in this plant, light cut column consists of three sections of structured packing using *Sulzer BX*. The three sections in the column are the condensing, rectifying, and stripping sections.

The feed from precut column enters between the rectification and stripping sections and flashes inside the column. Lighter components rise to the top and condense by direct contact with the cooled, down-flowing pump-around fluid in the condensing section. Part of the hot distillate returns to the column as reflux to the rectification section of the column. The remaining is cooled by tempered water in the lights cut product cooler before being sent to storage. The heavies collect in the bottom of the column and are circulated by a bottom pump through the reboiler. The net bottoms product is pumped to the middle cut column for further fractionation.

Process flowsheeting and plant simulation of the fractionation plant was carried out using HYSYS.Plant dynamic simulator based. The flowsheet developed in HYSYS environment can be seen in Figure 6. HYSYS.Plant does not support packed column and as such, the height equivalent theoretical plate (HETP) calculation was used. In the simulation, UNIQUAC activity model is selected for physical property calculations due to the facts that fatty acids (carboxylic acids) are polar component with non-ideal behaviour.

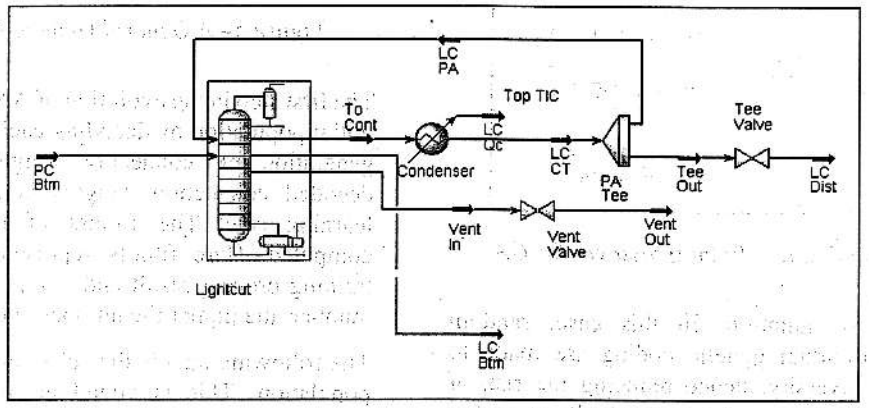


Figure 6 – Light Cut Column in PCOC Fractionation Plant

For plant model validation, comparison with actual plant dynamic was done using data collected from the plant DCS system. Only selected major stream and some state condition for light column such as temperature and pressure were compared. Minor discrepancies are accepted since the overall behaviour is what is important for this work. Simulation data was monitored to ensure no unreasonable outliers due to simulation errors. The simulation model developed adequately represents the selected distillation column since there is no major error.

ANN Model Development

Similar to other system identification approaches, model development for estimators involved several main considerations. First, a sensitivity analysis on the process was carried out. This was followed by carefully planned data generation for the model development tasks. Model development stages include specification of network architecture, selection of network topology, as well as model training and validation. Well-design identification efforts improve the validity of the resulting models. The procedures of developing ANN-GA estimator were similar to development of ANN-LM estimator. However, for the former, evolution was introduced in network training. All the development of ANN model endeavours were accomplished using MATLAB toolbox.

Sensitivity Analysis

Sensitivity analyses of light cut column were carried out within HYSYS.Plant environment for both open-loop and closed-loop systems. These analyses were carried out to provide better understanding on dynamic behaviour of the

ISBN:983-2643-15-5

process. In this case, step changes were imposed to various process inputs such as flow rate, feed temperature and feed composition. Their effects on process outputs such as column temperature, pressure and product composition were studied focussing on the sensitivity of the system.

Data Generation and Modification

Since the product quality variable cannot be measured on-line, a number of measured variables can be used to inferentially estimate the composition. The choice of suitable measurements were on process insights. In this case the results of sensitivity analyses become the main deciding factor.

Once the suitable variables were specified, data having sufficient information were generated form the HYSYS.Plant flowsheet. In order to develop sufficiently excited data for modelling purposes, the values of input variables were changed randomly. Ideally, the observations should cover the entire range of operating conditions. The data was then scaled using mean scaling method to prevent unnecessary domination of certain variables due to the difference in order of magnitude of actual process values.

Networks Architecture Specification

Architecture of ANN model includes type of networks, number of hidden layers as well as the number of hidden nodes. In this work, only feedforward network was considered. Only one hidden layer was utilised as it has come to general knowledge that an ANN with one hidden layer is sufficient to perform process mapping arbitrarily well. Another important stage in ANN model development is the selection of network topology. This is essentially the selection of the optimum number of hidden nodes or neurons. Usually, this number will increase with the complexity of system. In this study, 1-20 neurons were tested for the optimum network topology by trial and error approach. Decisions on the optimum topology were based on cross-validation, i.e., testing the trained model using a different set of data.

Training and Validation

In general, an ANN model undergoes training by successive presentation of input-output pairs and adjustment of weights to minimise the deviation of the network estimations from specified target values. This can be achieved using any suitable optimization algorithm. Levenberg-Marquardt (LM) method that is one of the backpropagation techniques was adopted for this work.

Evolution of Connection Weights

Network training is a minimization of an error function such as MSE, by iteratively adjusting connection weights. GA can be used effectively in the evolution to find a near-optimal set of connection weights globally without computing gradient information, thus eliminating the weakness of gradient based methods. The evolutionary approach to weight training in ANN consists of two major phase. The first phase is to decide the representation of connection weights in the form of binary or not. The second phase involves evolutionary process by GA.

Results and Discussion

Multi-input multi-output (MISO) model was chosen to for the development of an estimator for the C-12 composition in distillate stream. Eight variables as well as their one-step delayed signals were fed as the model inputs. These input variables include reflux flowrate, recycle flowrate, top column temperature, feed temperature and four tray temperatures. A total of 1000 data was generated using these model inputs for modelling purposes. The data was divided into training and validation set. In the network architecture specification, the optimum neuron was decided based on the minimum validation mean square error (MSE). It was found that the C-12 composition could be estimated accurately using a network with 8 hidden neurons. In this case, the network is trained using LM. The transfer functions used in the network are log sigmoid and linear for hidden layer and output layer respectively. The performance of both ANN-LM and ANN-GA model were considered in offline and online estimation.

Offline Estimation

Both ANN-LM and ANN-GA models were tested on four set of different range data for model evaluation. The results are summarised in Table 1 and plotted in Figure 7. Data C were used as training and validation data in model development.

Table 1 – Offline Estimation Results

	ANN-LM	**ANN-GA**
TrainMSE	8.54E-05	2.76E-04
ValMSE	1.65E-04	3.19E-04
CrossMSE		
DataA	1.51E-04	1.40E-04
DataB	9.34E-04	7.36E-04
DataC*	1.25E-04	2.97E-04
DataD	9.55E-05	5.61E-05
Average	**3.26E-04**	**3.07E-04**

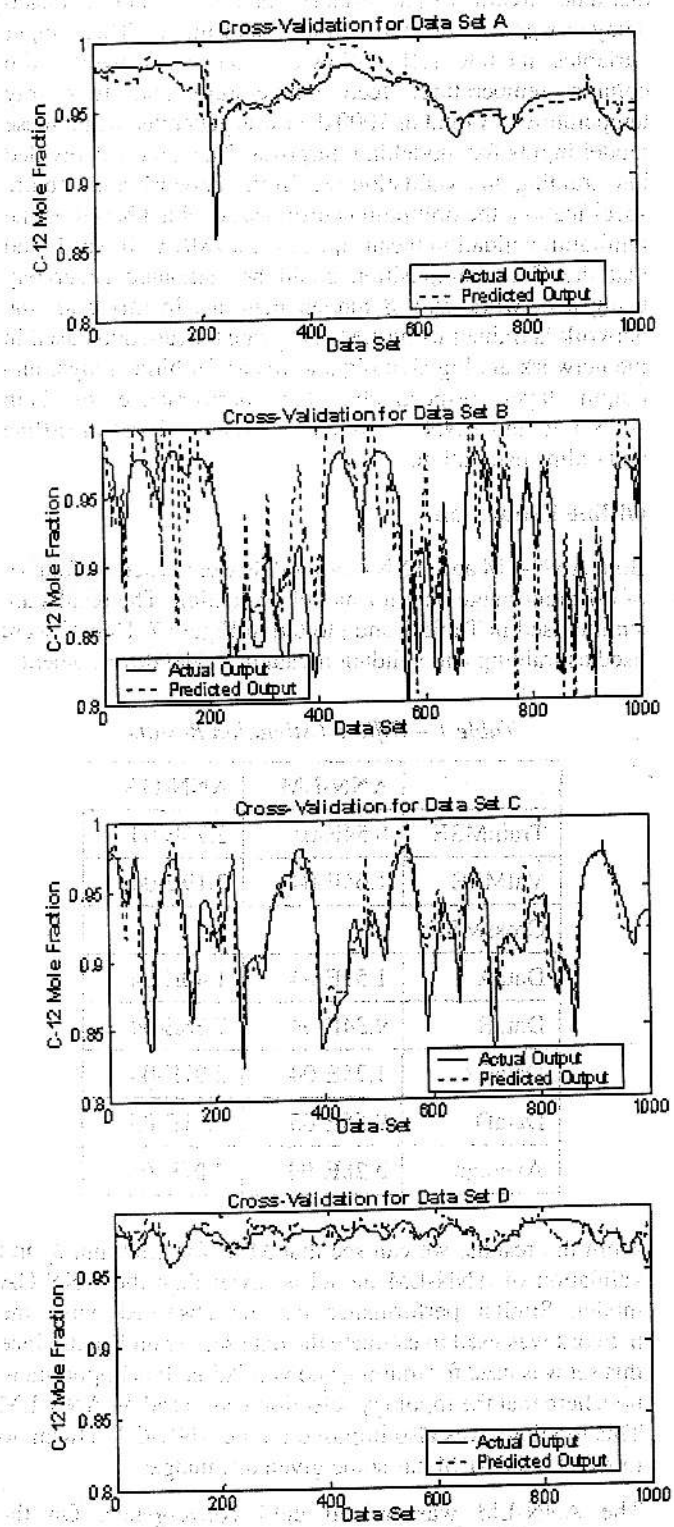
From the results, we can see that MSE for the training and validation of ANN-LM model is lower than the ANN-GA model. Similar performance can be observed when the network was used to estimate the data scatter on DataC since this set was used for training and validation. It is important to note here that the topology selection was based on ANN-LM. This topology was also imposed on the ANN-GA. Owing to this fact, ANN-LM has some given advantages.

The ANN-LM was trained until convergence. On the contrary, ANN-GA was trained only until some specified number of generations (i.e., iterations). In order to reduce

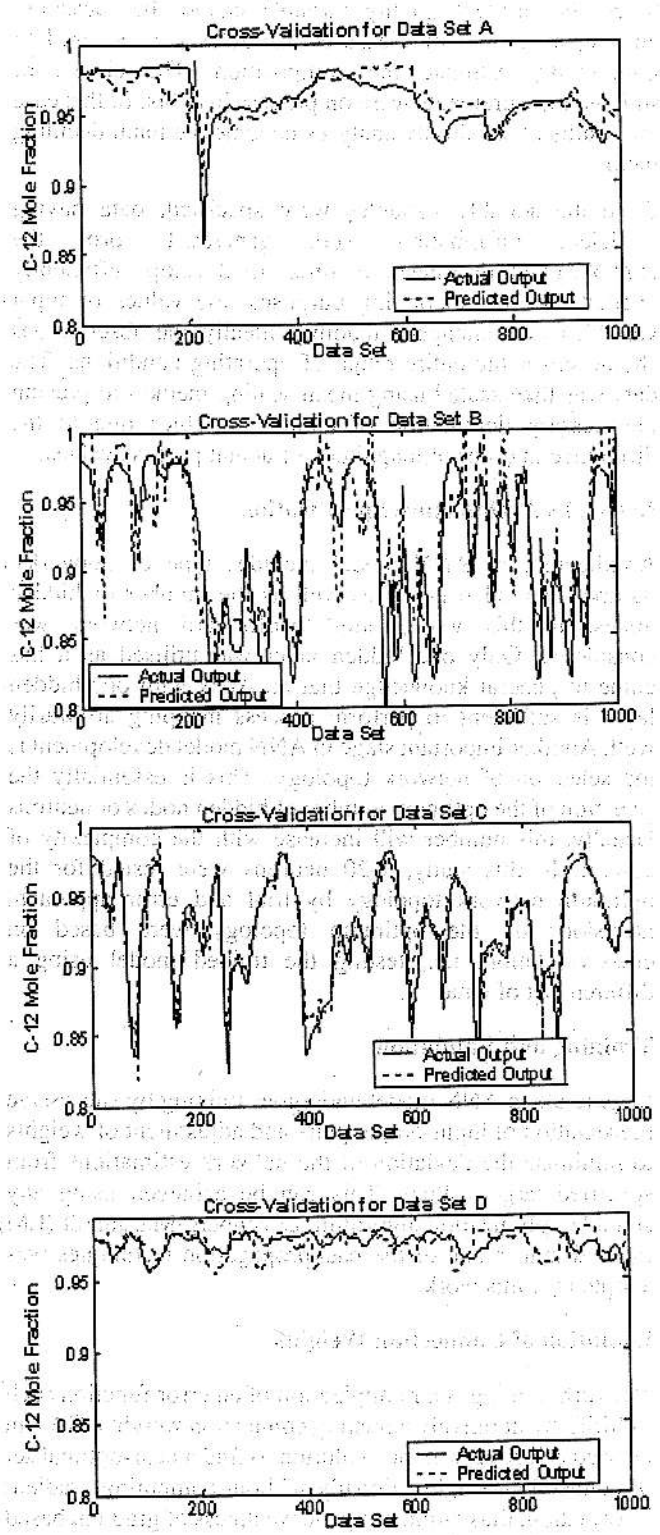
ISBN:983-2643-15-5

simulation time, the number of generation for evolution in GA cannot be too large. There is a trade-off between number of generation and simulation time for ANN-GA model. In this work, the number of generation was set to 150 and this was believed to be able to train the network to sufficient accuracy.

Although ANN-LM was well trained, the overall performance of ANN-GA is better when evaluated agains other sets of data. This proved that ANN trained by GA can generalise better in different pattern of data and can better cope with different operating condition, thus making the model more robust. ANN-LM model is poor in extrapolation. This is shown by poor estimation of DataB, which has bigger range than DataC.



(a) Cross-validation Using ANN-LM



(b) Cross-validation Using ANN-GA

Figure 7 – Performances of ANN Model in Offline Cross-validation

ISBN:983-2643-15-5

Online Estimation

Adaptations of network weights in both ANN-LM and ANN-GA model were introduced. The strategy was to activate the retraining process using once the estimation error was exceeding the predefined limit. The retraining allowed automatic update of the connection weights (including bias) based on the same network architecture to take into considerations new conditions. Here, the network restructuring was not considered and retraining of initial network was assumed to be able to provide improvements to estimation properties.

ANN-LM and ANN-GA models were tested using two sets of data: normal operating condition with small disturbances imposed and operating conditions with big disturbances. The online estimation results are shown in Table 2 and are illustrated in Figure 8.

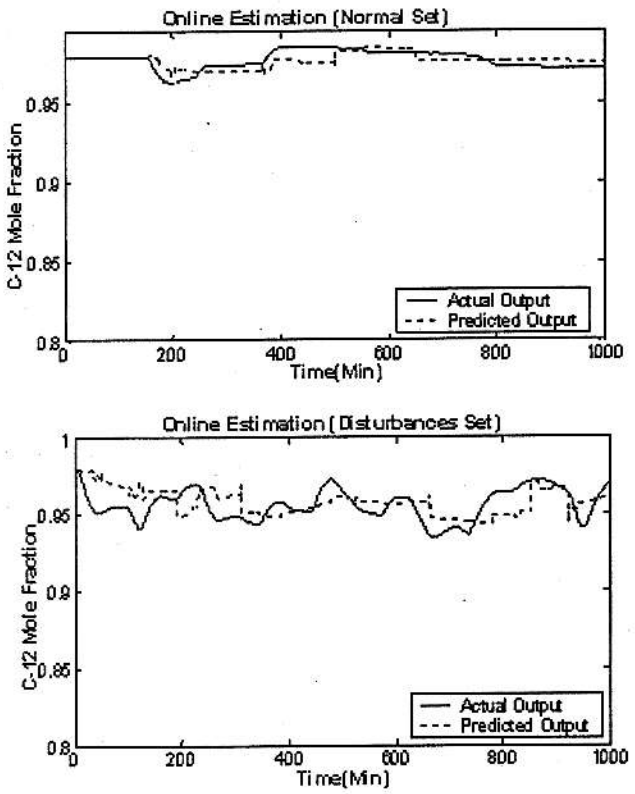
From the results, it is clear that ANN-GA model can perform better for both operating conditions compared to ANN-LM model. Although both models can follow the trend during the estimation, ANN-GA model provides better accuracy.

The poor performances of ANN-LM model may be due to the local initial weights tend to lead the solution converge at local minima.

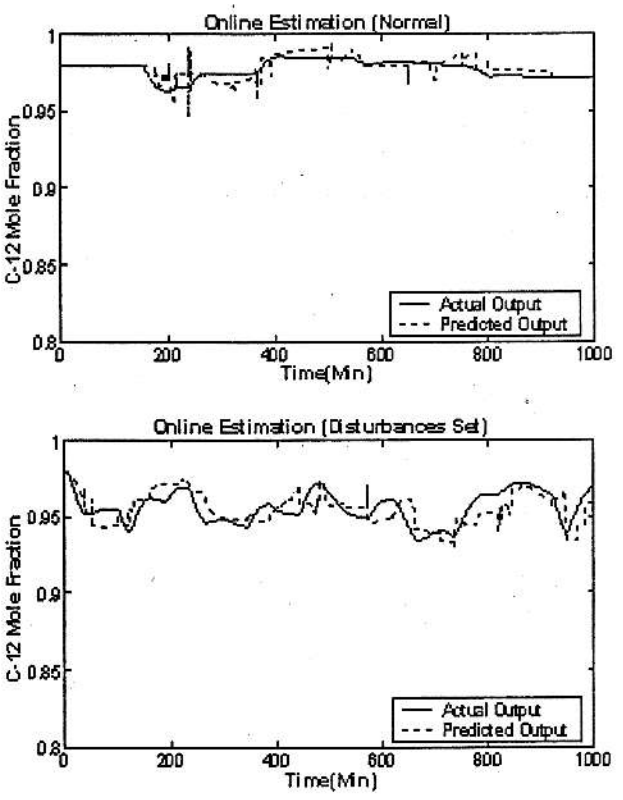
Better results were obtained for ANN-GA model. This suggests that by using GA to retrain network, the updated connection weights were able to adapt to new operating conditions. The ANN-GA model was gaining adaptability to dynamic nonlinear environment.

Table 2 – Online Estimation Results

	Normal		Disturbances	
	ANN-LM	*ANN-GA*	*ANN-LM*	*ANN-GA*
MaxE	0.0112	0.0194	0.201	0.0245
MinE	-0.0132	-0.0257	-0.0302	-0.0259
Range	0.0243	0.0451	0.0503	0.0504
Trend	Yes	Yes	Yes	Yes
MSE	**2.24E-05**	**1.80E-05**	**1.27E-04**	**7.95E-05**



(a) Adaptation Using ANN-LM



(b) Adaptation Using ANN-GA

Figure 8 – Performances of ANN Model in Online Adaptation

Despite this encouraging observation, there are still questions in implementing ANN-GA model in a real plant. Since the ANN-GA model only concentrates the training algorithm using predefined network architecture, the selection of initial network architecture is particularly important. Different types of networks such recurrent networks with auto correlation information may also be considered although making training procedure more complicated.

Another issue is regarding the training data used for model adaptation. Commonly, a good training data should cover a

ISBN:983-2643-15-5

wide range of operating conditions. This is difficult to establish in practice. One way of overcoming this issue is using initial connection weights that are trained under wide range of operating conditions, but adaptations are added to some tuneable parameters. For an example, new set of weights may act as tuning parameter and merging them with the initial weights by pre-specified percentage. In addition, large disturbances may also due to noise and signal errors. Hence, some form of filters may need to remove or to smooth out process signals. Appropriate filter design often leads to better robustness characteristics and hence makes the estimation scheme more practicable.

Conclusion

In this paper, we have demonstrated that better network training can be established using Genetic Algorithms. Introducing evolutionary mechanism in network training, ANN-GA performed better than ANN-LM either in offline or on-line implementation. ANN-GA was also demonstrated to be able to cope with different range of operating condition, thus improving the model robustness of ANN.

Acknowledgments

This project is funded by the Ministry of Science, Technology and the Environment through National Science Foundation Scholarships and IRPA research grant. Our heartiest appreciations are for everybody who has directly or indirectly contribute to the success of this project.

References

[1] Page, G.F., Gomm, J.B. and Williams, D. (1993). *Application of Neural Networks to Modelling and Control*. UK: Chapman &Hall

[2] Basheer , I.A. and Hajmeer M. (2000). Artificial Neural Networks: Fundamentals, Computing, Design, and Application. *Journal of Microbiological Methods*. 43: 3–31

[3] Sexton, R.S., Dorsey, R.E. and Johnson, J.D. (1998). Towards Global Optimization of Neural Networks: A Comparison of the Genetic Algorithm and Backpropagation. *Decision Support Systems*. 22: 171-185.

[4] Tham, M.T., Montague, G.A., Morris, A.J. and Lant, P.A. (1991). Soft-Sensors For Process Estimation and Inferential Control. *Journal of Process Control*. 1(1): 3-14.

[5] Ahmad, A., L.Y. Ling and T.S. Wong (2001). Neural Networks Estimators for Product Composition in a Palm Oil Fractionation Process. In Proceedings of PORIM International Palm Oil Conference 2001.

[6] Bhartiya, S. and Whiteley, J.R. (2001). Development of Inferential Measurements Using Neural Networks. *ISA Transactions*. 40: 307-323.

[7] Hunt, K.J., Sbarbaro, D., Zbikowski, R. & Gawthrop, P.J. (1992). Neural Networks for Control Systems – A Survey. *Automatica*. 28(6): 1083-1112.

[8] Ungar L.H., Hartman, E.J., Keeler, J.D. and Martin, G.M. (1996). Process Modeling and Control Using Neural Networks. *AIChE Symposium Series*. 92: 57-67.

[9] Hintz, K.J. and Spofford, J.J.(1990). Evolving a Neural Networks." In Proceedings of 5th IEEE International Symposiumon Intelligent Control. 1: 479-484.

[10]Montana, D. and Davis, L. (1989). Training Feedforward Neural Networks Using Genetic Algorithms. Proceedings of Eleventh International Joint Conference on Artificial Intelligence: 762-767.

[11]Yao, X. (1997). Global Optimization by Evolutionary Algorithms. In Proceedings of the 2nd AIZU International Symposium on Parallel Algorithms / Architecture Synthesis: 282-291.

[12]Fleming, P.J. and Purshouse, R.C. (2001). Genetic Algorithms in Control System Engineering. Research Report No. 789, University of Shffield.

[13]Branke, J. (1995). Evolutionary Algorithms for Neural Network Design and Training. Technical Report No.322, University of Karlsruhe, Institute AIFB.

ISBN:983-2643-15-5